# Final Exam

**CMSY-199, Spring 2011**

Circle the letter of the best response for each item.

1. What class is the root of the Java class hierarchy?
   (a) `java.Object`
   (b) `java.lang.Object`
   (c) `java.system.Object`
   (d) `java.default.Object`
2. Which of the following represents the *is a* relationship between classes?
   (a) Inheritance
   (b) Composition
   (c) Dependency
   (d) Realization
3. Which of the following represents the *has a* relationship between classes?
   (a) Inheritance
   (b) Composition
   (c) Dependency
   (d) Realization
4. The mechanism by which a superclass variable invokes an overriden method in a subclass is called
   (a) Abstraction
   (b) Encapsulation
   (c) Information Hiding
   (d) Polymorphism
5. Which of the following may not contain *any* method implementations?
   (a) abstract class
   (b) interface
   (c) subclass
   (d) superclass

6. Which of the following is *not* a runtime exception thrown by the JVM?
   (a) `ClassCastException`
   (b) `NullPointerException`
   (c) `SemicolonMissingException`
   (d) `ArrayIndexOutOfBoundsException`
7. The size (in bytes) of the primitive `char` type in Java is
   (a) 1
   (b) 2
   (c) 4
   (d) 8
8. What character encoding set does Java use to represent characters?
   (a) ASCII
   (b) EBCDIC
   (c) Unicode
   (d) UTF-8
9. A car dealership needs a program to store information about the cars for sale. For each car, they want to keep track of the following information: number of doors (2 or 4), whether the car has air conditioning, and its average number of miles per gallon. Which of the following is the best design?
   (a) Use one class, `Car`, which has three data fields: `int numDoors`, `boolean hasAir`, and `double milesPerGallon`.
   (b) Use four unrelated classes: `Car`, `Doors`, `AirConditioning`, and `MilesPerGallon`.
   (c) Use a class `Car` which has three subclasses: `Doors`, `AirConditioning`, and `MilesPerGallon`.
   (d) Use a class `Car`, which has a subclass `Doors`, with a subclass `AirConditioning`, with a subclass `MilesPerGallon`.
   (e) Use three classes: `Doors`, `AirConditioning`, and `MilesPerGallon`, each with subclass `Car`.

10. Consider the following Java classes:

```
public class DavidBanner
{
   public DavidBanner()
   {
      System.out.println("Mr. McGee, don't make me angry.");
   }

   public void speak()
   {
      System.out.println("You wouldn't like me when I'm angry.");
   }

}

public class IncredibleHulk extends DavidBanner
{
   public void speak()
   {
      System.out.println("Roar!");
   }
}
```

What is the output produced by the following statements?

```
DavidBanner david = new DavidBanner();
david.speak();
```

(a)
```
Mr.  McGee, don't make me angry.
You wouldn't like me when I'm angry.
```

(b)
```
Mr.  McGee, don't make me angry.
Roar!
```

(c)
```
You wouldn't like me when I'm angry.
```

(d)
```
Roar!
```

11. All subclasses of which class are considered unchecked exceptions?

(a) `AbstractException`

(b) `Exception`

(c) `GeneralException`

(d) `RuntimeException`

12. Which of the following is *not* one of the three stream objects associated with devices that Java creates when a program begins executing?

    (a) `System.in`

    (b) `System.out`

    (c) `System.err`

    (d) `System.exit`

13. What class from the `javax.swing` class is often extended to produce the top-level window of a GUI-based desktop application?

    (a) `JApplet`

    (b) `JFrame`

    (c) `JPanel`

    (d) `JTextField`

14. Given the following two constructors for the `Complex` class:

```
public Complex(double r, double i)
{
   this.real = r;
   this.imaginary = i;
}

public Complex()
{
   /* Insert line of code here */
}
```

    Which line of code could be inserted into the no argument constructor to make it create a `Complex` object with the real part and imaginary part both equal to 0?

    (a) `this();`

    (b) `this(0,0);`

    (c) `super(0,0);`

    (d) `return new Complex(0,0);`

15. What type of relationship exists when a class must implement the behavior specified in an interface?

    (a) Aggregation

    (b) Composition

    (c) Dependency

    (d) Realization

16. The `BasePlusCommissionEmployee` class is to be rewritten using an inheritance relationship rather than composition.

```
public class BasePlusCommissionEmployee
{
   private CommissionEmployee commissionEmployee;
   private double baseSalary;

   public BasePlusCommissionEmployee(String first, String last, String ssn,
      double sales, double rate, double salary)
   {
      commissionEmployee = new CommissionEmployee(first, last, ssn, sales, rate);
      baseSalary = salary;
   }
}
```

```
public class BasePlusCommissionEmployee extends CommissionEmployee
{
   private double baseSalary;

   public BasePlusCommissionEmployee(String first, String last, String ssn,
      double sales, double rate, double salary)
   {
      /* Insert line of code here */
      baseSalary = salary;
   }
}
```

Which line of code should be inserted to complete the rewritten six-argument constructor?

(a) `this();`

(b) `this(first, last, ssn, sales, rate);`

(c) `super(first, last, ssn, sales, rate);`

(d) `return new CommissionEmployee(first, last, ssn, sales, rate);`

17. What type of relationship exists between the classes `Chocolate`, `PeanutButter`, and `ReesesCup` if the `ReesesCup` class has member variables of type `Chocolate` and `PeanutButter`?

(a) Inheritance

(b) Composition

(c) Dependency

(d) Realization

18. Consider the following Java class:

```
 1 public class AbstractArt
 2 {
 3    public String name;
 4    public double value;
 5
 6    public String toString()
 7    {
 8       return String.format("name=%s value=%s ",name,value);
 9    }
10
11    public static void main(String args[])
12    {
13       AbstractArt pollock = new AbstractArt();
14       pollock.name = "No. 5, 1948";
15       pollock.value = 1.518E8;
16       System.out.println(pollock);
17    }
18 }
```

What is the output when the class is compiled and run?

(a) `name=No.  5, 1948 value=1.518E8`

(b) Compilation error on line 6

(c) Compilation error on line 13

(d) An exception is thrown at runtime

19. What must be done to prevent classes which extend the following class from modifying the values of the inherited fields?

```
public class PhysicalConstant
{
   public double SPEED_OF_LIGHT = 2.99792458e8;
   public double IDEAL_GAS_CONSTANT = 8.314472;
   public double PLANCKS_CONSTANT = 6.62606896e-34;
   public double AVOGADROS_NUMBER = 6.0221415e23;
}
```

(a) Add the modifier `final` to each field declaration

(b) Add the modifier `static` to each field declaration

(c) Add the modifier `protected` to each field declaration

(d) Nothing needs to be done

20. Consider the following Java class:

```
import java.io.*;

public class ExceptionCatcher
{
   public static void main(String args[])
   {
      String filename = "Foo.java";
      try
      {
         FileReader foo = new FileReader(filename);
      }
      catch(FileNotFoundException fnfe)
      {
         System.out.println("The file " + filename + " cannot be found.");
      }
      catch(Exception e)
      {
         System.out.println("An exception has occurred.");
      }
   }
}
```

If the file `Foo.java` does not exist, what is the output when the class is compiled and run?

(a) `The file Foo.java cannot be found.`

(b) `An exception has occurred.`

(c)
```
The file Foo.java cannot be found.
An exception has occurred.
```

(d) Compilation error

21. Which of the following has the items in the correct order for a valid Java source code file?

(a) import declarations, package declaration, class declarations

(b) package declaration, import declarations, class declarations

(c) package declaration, class declarations, import declarations

(d) class declarations, package declaration, import declarations

22. Given the following Java classes which are already compiled and in the classpath:

```
package car.japan;

public class Honda
{
   public static void printSlogan()
   {
      System.out.println("Honda: The Power of Dreams");
   }
}

package car.germany;

public class Volkswagen
{
   public static void printSlogan()
   {
      System.out.println("Volkswagen: Das Auto");
   }
}
```

What is the output when the following application is compiled and run?

```
 1 import car.japan.*;
 2 import car.germany.*;
 3
 4 public class Automobile
 5 {
 6    public static void main(String args[])
 7    {
 8       Honda.printSlogan();
 9       Volkswagen.printSlogan();
10    }
11 }
```

(a)
```
Honda:  The Power of Dreams
Volkswagen:  Das Auto
```

(b) Compilation error on line 1

(c) Compilation error on line 8

(d) An exception is thrown at runtime

23. Given the following Java class which is already compiled and in the classpath:

```
package cigar.cuba;

public class Cohiba
{
   public String toString()
   {
      return new String("Handmade from the finest tobacco available in Cuba");
   }
}
```

What must be done to make the following application compile and run?

```
import cigar.cuba.Cohiba;

public class CigarAficionado
{
   public static void main(String args[])
   {
      System.out.println(new Cohiba());
   }
}
```

(a) Nothing must be done to make the application compile and run

(b) Move the `CigarAficionado` class to the `cigar` package

(c) Make the `CigarAficionado` class a subclass of the `String` class

(d) Change the access modifier of the `Cohiba` class

24. What interface can be implemented to allow for event-handling capabilities to be associated with a `JButton` ?

(a) `ActionEvent`

(b) `ActionListener`

(c) `actionPerformed`

(d) `ActiontHandler`

25. The process by which an entire object may be written to or read from a file is called

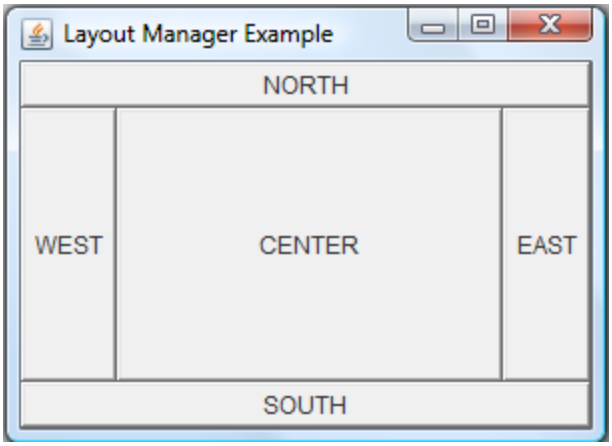
(a) Abstraction

(b) Encapsulation

(c) Realization

(d) Serialization

26. Which layout manager is depicted in the following figure?



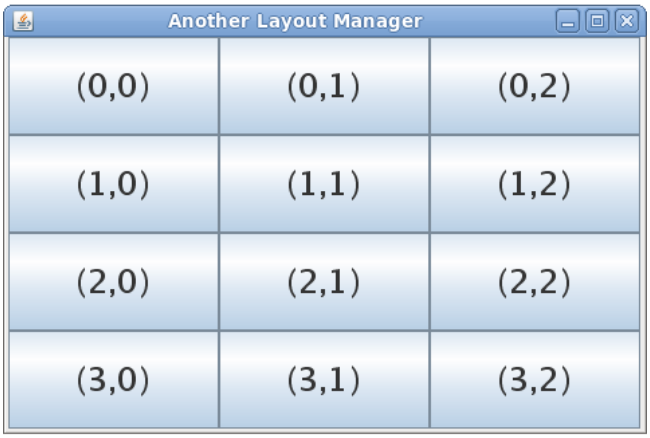

(a) `BorderLayout`

(b) `FlowLayout`

(c) `GridLayout`

(d) `DefaultLayout`

27. Which layout manager is depicted in the following figure?




(a) `BorderLayout`

(b) `FlowLayout`

(c) `GridLayout`

(d) `DefaultLayout`

28. Consider the following Java class:

```
package animal.mammal.marine;

public class Whale
{
   protected String getMantra()
   {
      return "Save the Whales";
   }
}
```

From which of the following can a method *not* access the `getMantra` method

(a) Inside the `Whale` class

(b) A subclass of the `Whale` class

(c) A class in the `animal.mammal.marine` package

(d) A class in the *default* package which doesn't extend the `Whale` class

29. What is the output when the following Java class is compiled and run?

```
 1 public class FrayedKnot
 2 {
 3    public void addChar(char c)
 4    {
 5       this.toString().concat(Character.toString(c));
 6    }
 7
 8    public static void main(String args[])
 9    {
10       FrayedKnot f = new FrayedKnot("I'm afraid not.");
11       f.addChar('!');
12       System.out.println(f);
13    }
14 }
```

(a) `I'm afraid not!`

(b) Compilation error on line 1

(c) Compilation error on line 10

(d) An exception is thrown at runtime

30. Given the following Java class:

```
import java.io.*;
import java.util.*;

public class TextFileReader
{
   public static void main(String args[])
   {
      try
      {
         /* Insert line of code here */
         Scanner s = new Scanner(f);
         while(s.hasNext()) System.out.println(s.nextLine());
      }
      catch(IOException ioe)
      {
         ioe.printStackTrace();
      }
   }
}
```

Which line of code could be inserted in order to read input from the file `textfile.txt`?

(a) `File f = new File("textfile.txt");`

(b) `FileInputStream f = new InputStream("textfile.txt");`

(c) `FileReader f = new Reader("textfile.txt");`

(d) `All of the above`

31. Which modifier can be used to cause a member variable to be ignored during the serialization process?

(a) `absolute`

(b) `public`

(c) `skip`

(d) `transient`

32. What class can be used to send data to any text-based stream?

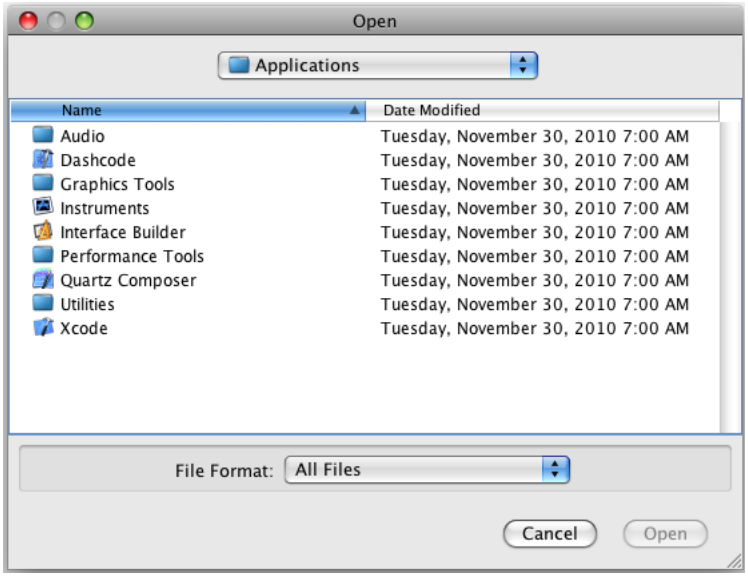
(a) `Formatter`

(b) `Outputter`

(c) `Scanner`

(d) `Streamer`

33. What Swing component is shown in the following figure?




(a) `JFileFormatter`

(b) `JFileStreamer`

(c) `JFileChooser`

(d) `JFileScanner`

34. What is the output when the following Java class is compiled and run?

```
import java.io.*;

public class IdentityVerifier
{
   public static void main(String args[])
   {
      Object s = new String("id");
      if (s instanceof String) System.out.println("Strings are Objects");
      if (s instanceof Serializable)
         System.out.println("Strings are Serializable");
   }
}
```

(a)
```
Strings are Objects
Strings are Serializable
```

(b)
```
Strings are Serializable
```

(c) Compilation error

(d) An exception is thrown at runtime

35. What is the output when the following Java class is compiled and run?

```
import java.io.*;

public class CornFlakes implements Serializable
{
   public static void main(String args[])
   {
      try
      {
         Object firstBowl = new CornFlakes();
         Object secondBowl = ((CornFlakes) firstBowl).cerealize();
         System.out.printf("%s %s %n",firstBowl.getClass().getName(),
                                    secondBowl.getClass().getName());

      }
      catch(Exception e)
      {
         e.printStackTrace();
      }
   }

   public Object cerealize() throws Exception
   {
      ObjectOutputStream output = new ObjectOutputStream(
         new FileOutputStream("breakfast.ser"));
      output.writeObject(this);
      ObjectInputStream input = new ObjectInputStream(
         new FileInputStream("breakfast.ser"));
      return input.readObject();
   }
}
```

(a) `Object Object`

(b) `CornFlakes Object`

(c) `Object CornFlakes`

(d) `CornFlakes CornFlakes`

36. Given the following Java classes which are already compiled and in the classpath:

```
public abstract class Dog
{
   public void speak()
   {
      System.out.println("Bark ");
   }
}

public class Achilles extends Dog
{
   public void speak()
   {
      System.out.print("RUFF ");
   }
}

public class Chloe extends Dog
{
   public void speak()
   {
      System.out.print("woof ");
   }
}
```

What is the output when the following application is compiled and run?

```
public class DogSpeak
{
   public static void main(String args[])
   {
      Dog pets[] = new Dog[2];
      pets[0] = new Achilles();
      pets[1] = new Chloe();
      for (Dog d : pets) d.speak();
   }
}
```

(a) `Bark Bark`

(b) `RUFF woof`

(c) Compilation error

(d) An exception is thrown at runtime

37. What is the output when the following Java classes are compiled and the `Simulation` application is run?

```
public class Simulation
{
   public static void main(String args[])
   {
      Building b = new Building();
      System.out.println("Building with 4 elevators created");
   }
}

public class Building
{
   private Elevator elevators[];

   public Building()
   {
      elevators = new Elevator[4];
      for (int i=0; i < 4; i++)
         elevators[i] = new Elevator();
      System.out.println("Elevators have been initialized");
   }
}

public class Elevator
{
   private int floor;

   public void setFloor(int f)
   {
      floor = f;
   }

   public int getFloor()
   {
      return floor;
   }
}
```

(a) `Elevators have been initialized`

(b)
```
Elevators have been initialized
Building with 4 elevators created
```

(c) Compilation error

(d) An exception is thrown at runtime

38. The following Java classes do not compile and run:

```
public class FootballTeam
{
   private String city;

   public FootballTeam(String city)
   {
      this.city = city;
   }

   /* Insert line of code here */

   public static void main(String args[])
   {
      FootballTeam winners = new Steelers();
      FootballTeam losers = new Ravens();
      System.out.println("Steelers vs. Ravens - 12/05/2010");
      System.out.println("Winners = " + winners.getClass().getName());
      System.out.println("Losers  = " + losers.getClass().getName());
   }
}

public class Steelers extends FootballTeam
{

}

public class Ravens extends FootballTeam
{

}
```

Which line of code could be inserted to make them compile and run?

(a) `public this(){}`

(b) `public super(){}`

(c) `public FootballTeam(){}`

(d) `public void FootballTeam(){}`

39. What is the output when the following Java class is compiled and run?

```
public class DivideByZero
{
   public static void main(String args[])
   {
      System.out.println("100 divided by 0 = " + (100/0));
   }
}
```

(a) `100 divided by 0 = 0`

(b) `100 divided by 0 = Infinity`

(c) Compilation error

(d) An exception is thrown at runtime

40. Given the following Java class:

```
public class Exam
{
   public static void main(String args[])
   {
      /* Insert line of code here */
   }

   public static void finish()
   {
      System.out.println("Congratulations! You have finished the exam.");
   }
}
```

Which line of code could be inserted to call the `finish` method?

(a) `super.finish();`

(b) `this.finish();`

(c) `Exam.finish();`

(d) None of the above